Literate Programming

Yaniv Lefel
Hagay Pollak

Part of a Course by Dr. Shuly Wintner.

1

Literate Programming

- Literate Programming (1983)
  - Donald E. Knuth
- Programming pearls – Literate Programming (1986)
  - Jon Bentley

``Beware of bugs in the above code; I have only proved it correct, not tried it.''

2

Introduction

- Improvements in programming methodologies - Structured programming.
  - Programs are more reliable
  - Easier to comprehend
- The next thing should be better documentation of programs.

4

Introduction – cont'

- We should consider programs to be *works of literature.*
- We use the title "Literate Programming".

5

Motivation

- Instead of instructing a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do.
- We introduce a new methodology: The WEB system.

6

Programming and weaving

- WEB - Three Letter Acronym.
- A program is like a WEB tangled and weaved, with relations and connections in the program parts. We express a program as a web of ideas.

7

The WEB System

- The WEB system was developed in Stanford University, by Knuth.
- The WEB system embodies the ideas of literate programming.
- WEB is a combination of
  - A document formatting language.
  - A program language.
- WEB does not make other languages obsolete, but rather enhances them.

8

The WEB System - cont

- TeX – document formatting language.
- Pascal – programming language.
  - Other languages can be used (and indeed are used).
- WEB language

I ... expect each member of the next generation of programmers to be familiar with a document and programming language.

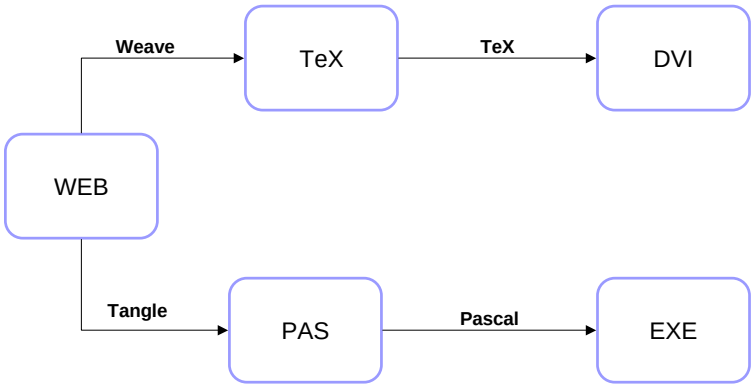
9

The WEB System – processes

- Weaving – producing a document describing the program clearly
  - Example: cob.w -> cob.tex -> cob.dvi
  - dvi - Device Independent binary description.
- Tangling - producing a machine executable program.
  - Example: cob.w -> cob.pas -> cob.exe
  - cob.exe – executable file.

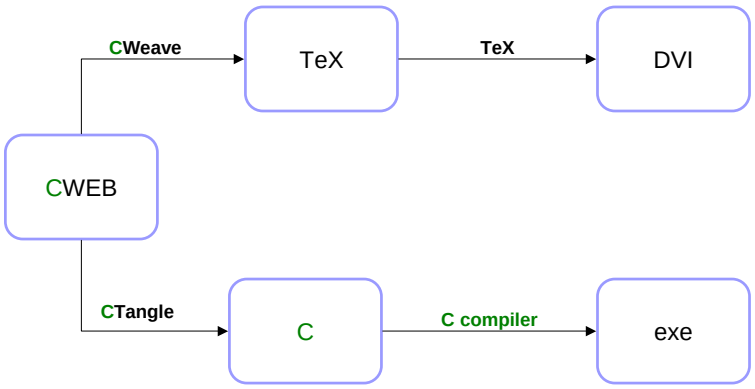
10

The WEB System – processes



11

The CWEB System – processes



12

Demo

- Hello World
- Fist

13

A few features of WEB

- WEAVE can parse program fragments.
  (Handle different languages ?)
- Macro usage.
- Statistics gathering
  - Keyword: *stat … tats*
- Overcoming Language misfeatures e.g. Pascal:
  - Local variables.
  - Samantics (semi-colon ;).

15

Left out of WEB

- Conditional macros
  - Evaluation of Boolean expressions cannot influence the output
  - Instead, we comment out optional code.

16

Portability

- WEB has already been transported to a wide variety of computers and OS.
- Based on the spread of TeX and Pascal/C compilers on various platforms.
  - Note: WEB is a framework and methodology, one can use other languages, such as c++, FORTRAN, …
- Deployment – based on open source, and on target compilation.

17

Portability – cont'

- Portability problems:
  - Character sets,
  - File naming conventions
  - Data packing
  - Etc.
- Change file – In addition to WEB source file, .ch files, e.g. TEX.CH (change file) contains changes required to customize TeX for a specific system.
  - New versions, can be updated.
  - Same master file TANGLE.WEB used by all.

18

Programming methodologies

- “Top-down” programming – top level description is given, and successively refined.
- “Bottom-up” - starting with the definitions of basic procedures and gradually building.

19

Top-down & bottom-up

pros & cons

- Top-down
  - You know where you are going.
  - You need to keep plans in your head.
- Bottom-up
  - You continually wield a more and more powerful pencil.
  - You need to postpone the overall program organization.

20

Opposing methodologies ?

- Are top-down and bottom-up programming opposing methodologies ?

21

Programs as webs

- No need to choose
- A program is best thought of as a web instead of a tree.
  - A hierarchical structure exists, but more important are structural relationships.
- The programmers task is to state the parts and relationships in whatever order is best for human comprehension.

22

Programs as webs – cont'

- A person reading a program is ready to comprehend it by learning its parts in a similar order in which it was written.

23

Programs as webs – cont'

- WEB allows expressing a program in a "stream of consciousness" order.
- Its always an order that makes sense.

24

Superior programs

- Traditional programming languages caused writing inferior programs
- WEB programs were better than programs in other languages.
  - A programmer subconsciously tries to get by with the fewest possible lines of code.
  - With WEB, it becomes natural to write better code, since it allows each part of the program have its appropriate size (maintaining readability).

25

Stylistic issues

- The reader should be able to grasp what is happening without being overwhelmed with detail.
- Section naming.
- Declaration location (use where needed not only where the language requires it).

26

Running WEB - time issues

- Running TANGLE on a WEB file, is the same time to compile a program.
- Program is converted into a booklet – increasing readability.
- Total time of writing and debugging a WEB program no greater than with regular programming language.
- Extra time in comments regained in debugging and maintaining.
- WEB language encourages a discipline.

27

Who does WEB suite?

- WEB language was not designed to suite everybody.
- WEB was intended for system programmers.
- You need to be able to deal with several languages simultaneously.
  - Compile time error (.w .tex .c).
  - Run time errors.
- I.e. WEB is best suited for computer scientists.

28

Conclusion

- A program may be thought of as a “web” of interconnected pieces.
- We want to explain each individual part and its relations to its neighbors.
- The typographic tools provided by TEX give us an opportunity to explain the local structure of each part.
- The programming tools provided by Pascal\C make it possible for us to specify the algorithms formally and unambiguously.
- By combining the two, we can maximize our ability to perceive the structure of a complex piece of software, and at the same time the documented programs can be mechanically translated into a working software system that matches the documentation.

29

Discussion

30